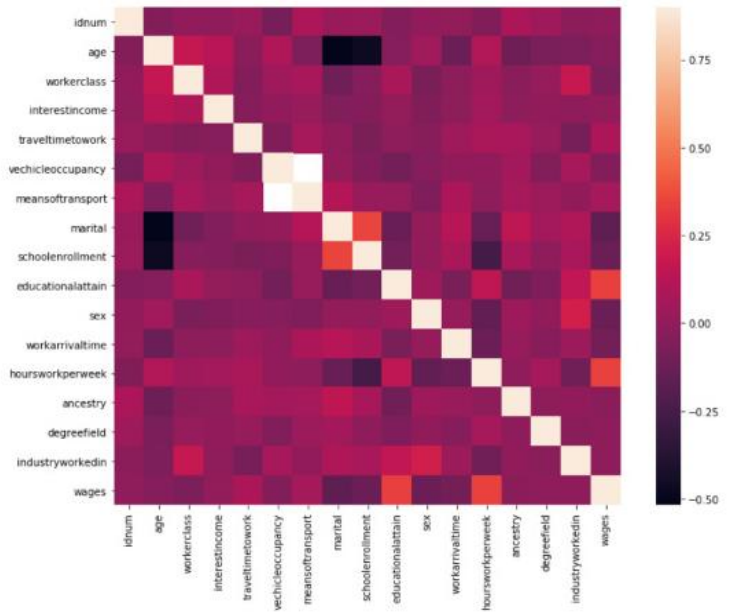
CS 6923: Machine Learning
Homework 5

Submitted by:
Nishika Chopra nc2259 N18598308
Karanpreet Singh Wadhwa ksw352 N13337853

REPORT

ABSTRACT
The data we were given was extremely skewed as well as there were a lot of zero and missing values so we started with filling these missing values. Together with this we also were analyzing features for feature selection and creation so that irrelevant features could be dropped. Then after preprocessing the data we trained the data by using. Linear regression and Random forest and on observing the efficiency of random forest we further improved our model by parameter tuning.

Data Exploration (EDA)
Missing values (Part 1)
We started with analyzing the data first for missing values. We compared which features had missing (NaN) values and what the count was for such values. In this way we identified the missing values so that we can do the next step of handling them.
Now the first intuition is to drop all the data with missing values. So we started by dropping the features which contained NaN values. Dejectedly it was observed that more than half of our data was gone. Thus dropping the missing values was a bad idea. Moving on, we filled in all the missing values with '0'. On performing a linear regression on the data we observed that the training had a very high mean square error thus we concluded that we had to first finely impute missing values then train our data.
Correlations and primary feature selection
Our first step in this problem was to understand the variables, assess them and do some basic EDA. We started with plotting the correlation matrix of the train data-set.
Some basic takeaways from the correlation matrix are as follows:



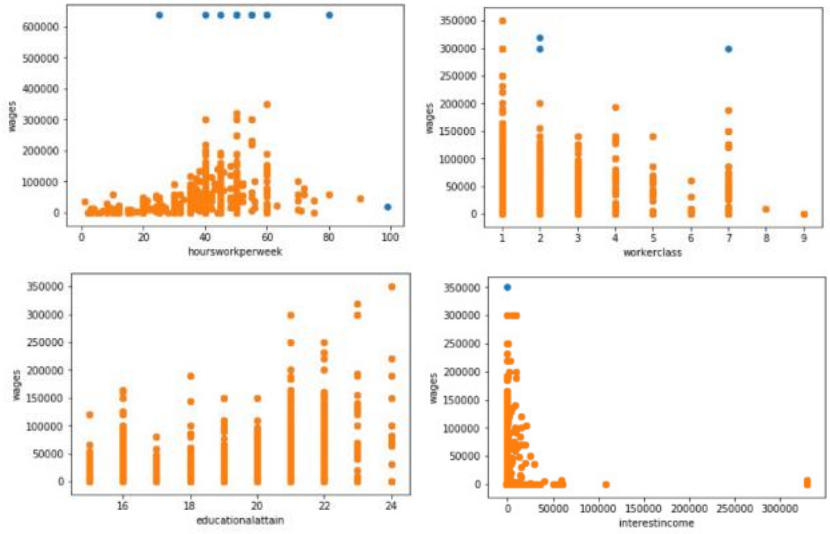
1. In the heatmap , all white squares indicate high correlation between the corresponding variables
2. 'vehicaloccupancy' and 'meansoftransport' show high correlation and are aligned with our intuitive thinking as well. This indicates multicollinearity and thus we remove one of them. If we compare the number of missing values in the two, we see that 'vehicaloccupancy' has higher number of missing values than 'meansoftransport'. Thus we retain 'meansoftransport' and drop 'vehicaloccupancy'.
3. We should keep in mind two other variables-'educationattain' and 'hoursworkperweek' that seem to have good correlation with 'wages'

A regression analysis separates the relationship between each independent variable (x) and the target/dependent variable (y). In the final equation, a regression coefficient gives the average change caused in the target variable for a unit change in the corresponding independent variable, given that all the other independent variables are constant.

Now in a case, when independent variables are correlated (this condition is multicollinearity), it indicates that changes in one variable would also cause shifts in another variable. The stronger this correlation is, higher would be the difficulty to change one variable without changing another. This makes it difficult for the regression model to estimate the relationship between each independent variable and the dependent variable independently. This basically leads to 3 major problems, which are related to each other:
1. Unreliable explained variance due to predictors
2. The coefficients become highly sensitive to small changes in the model and the presence of other independent predictors.
3. It reduces the precision of the equation coefficients, thus reducing the performance of our regression model (unreliable p-values to determine the statistical significance of the independent variables).
Thus we preprocess our data before training.

Outlier Treatment

Regression as an algorithm, tries to fit the best line through the data points. To find the best fit function, it tries to accommodate for all points which makes it very sensitive to outliers.
We employed a heuristic method of treating the outliers so that we could see and control the data points we are treating. Some variables we paid attention to for outlier treatment were: hoursperweek, workerclass, educationattain and interestincome.
Here the blue dots indicate the examples that we consider of being outliers hence we have removed them.



Missing Values Treatment (Part 2)

There are several ways to remove missing values and imputing them with viable substitutes. At first, we checked for missing values in all the variables in training data. This table will help us prioritize the order in which we want to tackle each of these variables.

idnum	0
age	0
workerclass	378
interestincome	0
traveltimetowork	576
vechicleoccupancy	775
meansoftransport	543
marital	0
schoolenrollment	0
educationalattain	0
sex	0
workarrivaltime	576
hoursworkperweek	465
ancestry	0
degreefield	766
industryworkedin	378
wages	0
dtype: int64	

Imputing missing values with numerical alternatives

Since we had tried two approaches to fill the missing data now we tried a third approach. After a thoughtful insight we filled the missing values as following:

1. Workerclass: age<16 -> category 0 and rest were assigned as category 10
2. Workarrivaltime: workerclass=0,9 -> category 0 and rest were assigned as category 286

3. Industryworkedin: workerclass=0,9 -> category 0 and rest were assigned as category 11
4. All other features we replaced the missing values with 0 category

Final Steps in Data Preparation

Treatment of categorical and ordinal variables

After all the features were created, we used One-Hot encoding and Binary encoding to treat the features. But in One-Hot encoding the number of columns came out to be very large due to which there was a chance of overfitting so we went with Binary encoding which is analogous to One Hot but with lesser columns. In this technique, first the categories are encoded as ordinal, then those integers are converted into binary code, then the digits from that binary string are split into separate columns.

Model creation and Training

REMARK: Since there were no labels in test set we divide the training set into training and validating set and use this validation set to check the mean square error of our model.

After careful consideration we used two methods/models to train our data, Linear regression and Random Forest. The reasons are as follows:

1. Whenever we encounter a regression problem our first instinct is to go with linear regression. From our experiences and learning, linear regression is a basic model that gives us the measure of how linearly separable the data is as well as we get an intuition of which method we can apply to get accurate results. We assess the output of the data to make a decision if more complex model is required or not.
2. Random Forest is considered as a very handy and easy to use algorithm, because it's default hyperparameters often produce a good prediction result. The number of hyperparameters is also not that high and they are straightforward to understand. One of the big problems in machine learning is overfitting, but most of the time this won't happen that easy to a random forest. That's because if there are enough trees in the forest, it won't overfit the model. It is very good at handling tabular data with numerical features, or categorical features with fewer than hundreds of categories. Unlike linear models, random forests can capture non-linear interaction between the features and the target. So that's why we would take Random forest Regressor as our second Model.

```
RandomForset regresion with 20 KFlod
RMSE for Kfold 1 : 25603.067492
RMSE for Kfold 2 : 28676.148242
RMSE for Kfold 3 : 27073.030825
RMSE for Kfold 4 : 27429.154320
RMSE for Kfold 5 : 22416.907096
RMSE for Kfold 6 : 29678.837740
RMSE for Kfold 7 : 22783.744188
RMSE for Kfold 8 : 25293.564246
RMSE for Kfold 9 : 46031.848011
RMSE for Kfold 10 : 21357.475954
RMSE for Kfold 11 : 21205.722272
RMSE for Kfold 12 : 44539.559302
RMSE for Kfold 13 : 38966.782802
RMSE for Kfold 14 : 36243.682313
RMSE for Kfold 15 : 33475.650172
RMSE for Kfold 16 : 23340.096114
RMSE for Kfold 17 : 27951.821735
RMSE for Kfold 18 : 21193.429309
RMSE for Kfold 19 : 28337.011526
RMSE for Kfold 20 : 21684.558478
RMSE_average_KFOLDS : 28664.104606819466
```

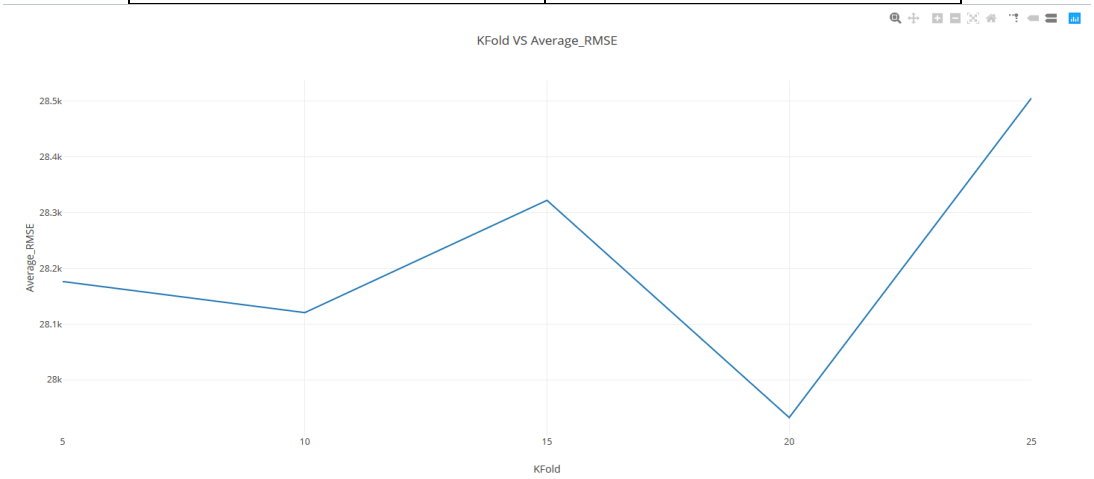
```
Linear regresion with 20 KFlod
RMSE for Kfold 1 : 41997.544577
RMSE for Kfold 2 : 31972.103090
RMSE for Kfold 3 : 31246.330008
RMSE for Kfold 4 : 45757.468885
RMSE for Kfold 5 : 32235.447065
RMSE for Kfold 6 : 20182.024020
RMSE for Kfold 7 : 16971.316404
RMSE for Kfold 8 : 35209.116571
RMSE for Kfold 9 : 21446.388440
RMSE for Kfold 10 : 27062.794014
RMSE for Kfold 11 : 23447.636175
RMSE for Kfold 12 : 31341.193620
RMSE for Kfold 13 : 33375.172052
RMSE for Kfold 14 : 33902.104370
RMSE for Kfold 15 : 34621.460600
RMSE for Kfold 16 : 35091.575619
RMSE for Kfold 17 : 24230.511743
RMSE for Kfold 18 : 21506.533436
RMSE for Kfold 19 : 26060.948441
RMSE for Kfold 20 : 34998.946341
RMSE_average_KFOLDS : 30132.830773552338
```

Since linear regression was a basic model to test the complexity and linearity of data we noted our observation and on finding the need for a more complex model we chose random forest as our main model. We also observed that random forest was performing better than linear regression and we decided to pursue it further.

Random Forest Model

For KFold

KFold	Average_RMSE
5	28176.36
10	28120.56
15	28321.9251
20	27932.43049
25	28505.1591



As 20 Fold is providing less Average_RMSE so we would take it as number of KFold!

Parameter Tuning

There are 2 parameters to be tuned. Namely, N_estimator and Max_depth.
N-estimator: Max number of Random forest Trees
Max-depth: Max depth of each tree

For KFold =20

Max-depth	N-estimator	Average_RMSE
2	30	31153.41121
3	30	29911.25539
5	30	28609.29443
None(As max as it can)	30	28020.9075
2	50	31218.9991
3	50	29832.9302
5	50	28202.6011
None(As max as it can)	50	27932.43049

As Average_RMSE is less for KFold =20, Max_depth =None and N_estimator=50. So we would choose these value as our parameters for predicting on test set.

REFERENCES

[1] https://towardsdatascience.com/data-preparation-and-preprocessing-is-just-as-important-creating-the-actual-model-in-data-sciences-2c0562b65f62
[2] https://www.datacamp.com/community/tutorials/categorical-data
[3] https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd
[4] https://anaconda.org/conda-forge/category_encoders
[5] https://machinelearning-blog.com/